Unidad 5- Estructuras de control

Las estructuras de control en programación son herramientas fundamentales que permiten a los programadores controlar el flujo de ejecución de un programa. Estas estructuras determinan cómo se comportará un programa bajo diferentes condiciones y situaciones. En esencia, son las reglas que controlan cómo un programa avanza de un paso al siguiente.

Hay dos tipos principales de estructuras de control:

1) Estructuras de control condicionales:
    a) If/else: Permite tomar decisiones basadas en condiciones. Si se cumple una condición, se ejecuta un bloque de código; de lo contrario, se ejecuta otro.
    b) Switch: Se utiliza para manejar múltiples casos posibles y ejecutar diferentes bloques de código según el valor de una variable.
2) Estructuras de control iterativas (bucles):
    a) Bucles for: Se repiten un número específico de veces, basado en una variable de control.
    b) Bucles foreach: Se repiten un número específico de veces, basado en la cantidad de elementos de una colección (generalmente arreglos).
    c) Bucles while: Se ejecutan mientras se cumpla una condición específica.
    d) Bucles do-while: Similar al bucle while, pero garantiza que el bloque de código se ejecute al menos una vez.

Estructura de control if/else

La estructura if/else se puede aplicar sin el apartado de else, indicando una condición (que tiene que ser boolean) y un código dentro de las llaves del bloque

```php
<?php
    if (condition) {
        # code...
    }
```

Funciona de manera similar a la hora de utilizar la instrucción else. En caso de que la condición de un valor **true** se realizarán las acciones dentro del bloque if. En caso contrario, se realizarán las acciones del bloque else

Aquí se presenta un ejemplo con operadores relacionales

Los operadores relacionales son los siguientes:

1) '<' significa menor que. Retorna true si el primer valor es menor que el segundo
2) '<=' significa menor o igual que. Retorna true si el primer valor es menor o igual que el segundo
3) '>' significa mayor que. Retorna true si el primer valor es mayor que el segundo
4) '>=' significa mayor o igual que. Retorna true si el primer valor es mayor o igual que el segundo
5) '==' significa igual que. Retorna true si el primer valor tiene igual valor que el segundo
6) '===' significa idéntico que. Retorna true si el primer valor tiene igual valor y tipo de dato que el segundo
7) '!=' significa distinto que. Retorna true si el primer valor tiene valor distinto que el segundo
8) '!==' significa no idéntico que. Retorna true si el primer valor tiene valor distinto o tipo de dato distinto que el segundo

Operadores lógicos

Los operadores lógicos nos permiten operar con múltiples valores booleanos en una operación. Se utilizan para obtener una condición extendida con más de una restricción
Estos son:

1) &&: Operador binario and que nos retorna true si ambas condiciones son true
2) | | : Operador binario or que nos retorna true si al menos una de las condiciones es true
3) xor : Operador binario xor que nos retorna true si solo una de las condiciones es true
4) ! : Operador unario not que nos retorna true si el valor de la condición referida es false y viceversa

```php
<?php
    $numero1 = 3;
    $numero2 = 4;
    if ($numero1 > 0 && $numero2 > 0) {
        echo "Hola";
    }else {
        echo "Chau";
    }
```

If como operador ternario

Tenemos una forma alternativa de realizar una estructura if/else de manera reducida
En este caso si se cumple la condición se aplica el valor de la izquierda de los “:” en caso contrario se aplica el valor de la derecha

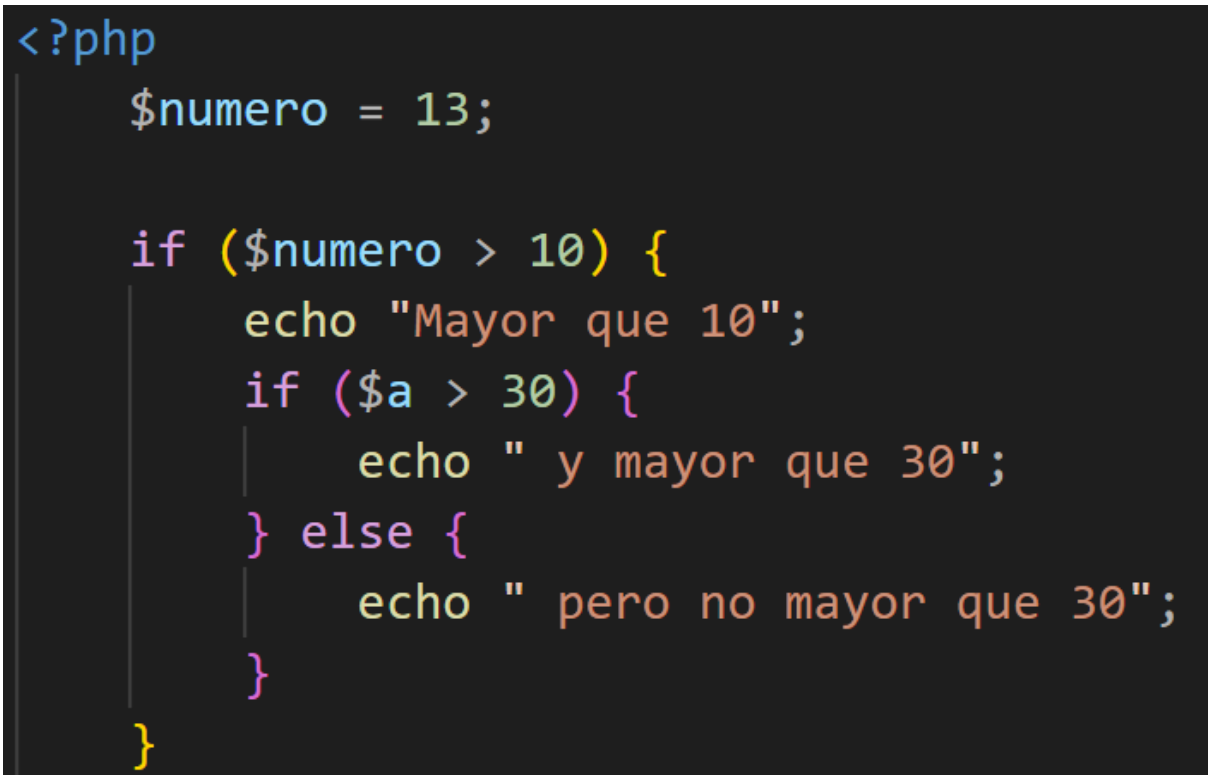
```php
<?php
    $numero1 = 13;

    $resultado = $numero1 < 10 ? "Hola" : "Chau";
    echo $resultado;
```

En el ejemplo anterior se mostrará “Chau”

If/else anidado

En este apartado lo que podemos realizar es incluir una estructura if dentro de otra. En este caso se deberá seguir el orden de ejecución de las anidaciones de manera escalonada



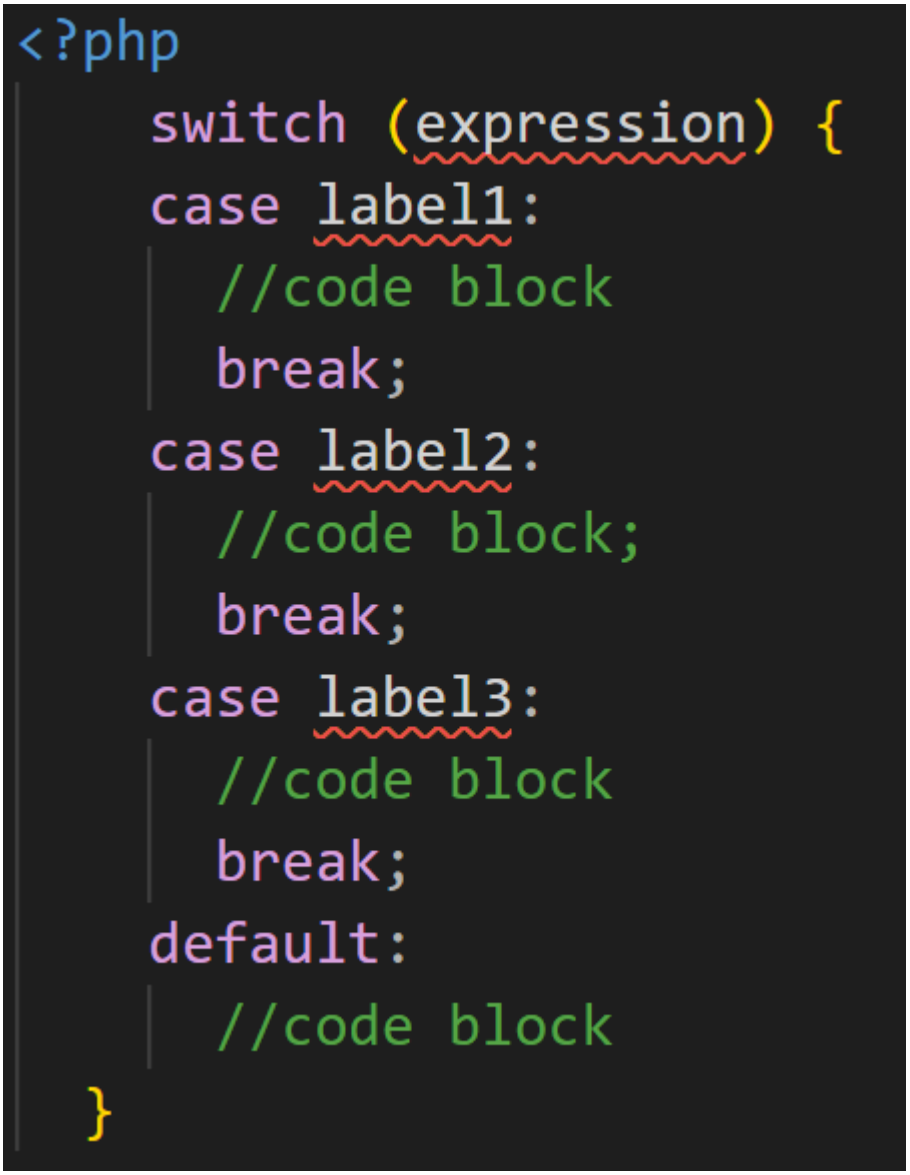
```php
<?php
    $numero = 13;

    if ($numero > 10) {
        echo "Mayor que 10";
        if ($a > 30) {
            echo " y mayor que 30";
        } else {
            echo " pero no mayor que 30";
        }
    }
```

En este caso se mostrará el mensaje “Mayor que 10 pero no mayor que 30”

Estructura switch

La estructura switch nos permite decidir en base al valor de una expresión, a cual de los bloques de código vamos a llamar para su ejecución. Cada bloque culmina con la expresión break, excepto default que será el código que se ejecutará cuando la expresión inicial no cumpla con ninguno de los bloques anteriores.



Un ejemplo del uso de switch es el siguiente

```php
<?php
    $verano = "enero";

    switch ($verano) {
        case "diciembre":
            echo "Estamos al comienzo del verano";
            break;
        case "enero":
            echo "Estamos en la primera mitad del verano";
        break;
        case "febrero":
            echo "Estamos en la segunda mitad del verano";
        break;
        case "marzo":
            echo "Estamos a fines del verano";
            break;
        default:
            echo "No estamos en verano";
    }
```

Bucle for

La estructura for nos sirve para iterar sobre un rango de repeticiones. Está compuesto por una expresión de inicio, una expresión condicional y una expresión de incremento/decremento (solo una de las dos). En este ejemplo el código contenido se repite 10 veces

```php
<?php
    for ($i=0; $i < 10; $i++) {
        # code...
    }
```

Bucle foreach

El bucle foreach lo hemos visto con anterioridad y nos sirve para recorrer colecciones de datos pasando por cada uno de sus elementos

```php
<?php
    $nombres = array("Thor", "Loki", "Odin");

    foreach ($nombres as $i) {
        echo "$i <br>";
    }
```

Bucle while

El bucle while se repite mientras que la condición establecida sea true

```php
<?php
    $numero=1;
    while ($numero <= 10) {
        echo $numero;
        $numero++;
    }
```

Bucle do…while

El bucle do…while se comporta de manera similar a while excepto por el hecho de que las instrucciones del bloque se ejecutan al menos una vez

```php
<?php
    $numero=1;
    do {
        echo $numero;
        $numero++;
    }while ($numero <= 10);
```